World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(REVIEW ARTICLE)

Check for updates

Evaluating the impact of Devsecops on software quality: A systematic review and empirical study

Gopinath Kathiresan *

Senior Quality Engineering Manager, CA, USA.

World Journal of Advanced Research and Reviews, 2022, 14(01), 644-653

Publication history: Received on 20 February 2022; revised on 22 April 2022; accepted on 25 April 2022

Article DOI: https://doi.org/10.30574/wjarr.2022.14.1.0267

Abstract

Security throughout the software development lifecycle is the prime concern in the current software engineering, thus giving rise to a natural evolution, DevSecOps, from DevOps. It, from the very outset of development, integrates security into the whole process and ensures its way into every phase of the software delivery pipeline, as opposed to the earlier paradigms of software development that would always consider security as an afterthought. Using a systematic literature review and an empirical study, this research investigates the influences of DevSecOps practices on quality, security, and organizational factors impacting software. The findings indicate that DevSecOps dramatically improves the security of software because it allows vulnerabilities to be identified and mitigated early on, thus redounding to the improved reliability of code while also reducing the attack surface. Automation here uses force to enable continuous security monitoring, automated vulnerability scanning, and rapid incident response. Continuous Integration and Continuous Deployment (CI/CD) pipelines promote smooth embedment of security checks into the software development workflow, thus reducing the possibility of their being introduced at later stages.

However, thus far, most challenges abound its adoption. Many organizations face the challenges with complexity of security tool; they have a steep learning curve for developers implementing DevSecOps; and there is often limited existence of security expertise in development teams. Finding a balance between conflicting issues such as trade-off security, on the one hand, and speed and agility that are the essence of DevOps on the other, continues to be a challenge because, on the one hand, security can slow down development cycles. Resistance to change and the culture of organizations are further barriers to the complete adoption of DevSecOps.

Therefore, the recommendations for addressing such problems include industry best practices like automated security testing, regulatory compliance, and culture fostering awareness on security. Organizations should also invest in training programs for developers with respect to security skills and create incentives for collaboration among development, security, and operations. Moreover, AI-based security automation promises to open an avenue through which security efficiency can be improved at high speed of detection of threats and by reducing manual efforts in security testing.

Keywords: Dev SecOps; Software security; Automation; Continuous monitoring; Security-aware culture; AI-driven security

1. Introduction

1.1. Background of DevSecOps

The demand for rapid development and deployment of software is on the rise, which has made organizations go to DevOps, a methodology integrating development and operations. It tends to create efficiencies, agility, and continuous

* Corresponding author: Gopinath Kathiresan

delivery within organizations. In essence, DevOps helps organizations streamline workflows and automate repetitive processes in order to accelerate the release of software updates. Most traditional approaches towards DevOps have emphasized speed and operational efficiency rather than adequately addressing security issues. That normally results in a sustained increase in vulnerabilities, overdue security patches, and reactive approaches to threat mitigation (Myrbakken & Colomo-Palacios, 2017).

The organization has evolved with developing a proactive approach towards security, which an organization realizes to be vital because speed in development should be maintained. This combination of qualities has given birth to DevSecOps, which are tactics that modifies DevOps with infusing security practices at every key stage of software development lifecycle (SDLC). Unlike traditional security models where security checks are carried out towards the end of the development processes, DevSecOps has automated security tools, policies, and best practices embedded from initial planning throughout the deployment and beyond (Cui, n.d.).

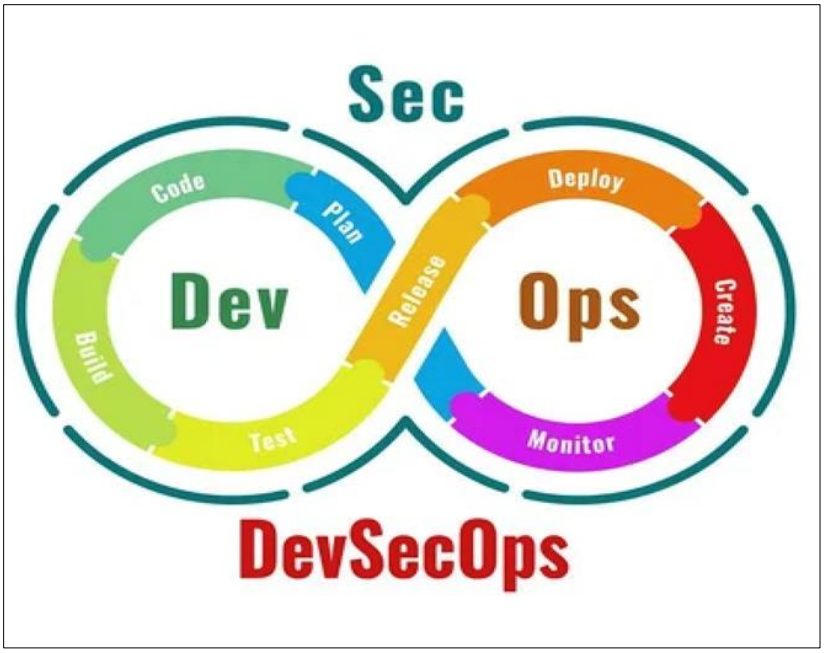

Figure 1 What is a DevSecOps and its importance to the software company

Embedding security in DevOps pipelines will lead DevSecOps to nurture a culture that concentrates on security-first in as much as it reduces the risk that can easily be mentioned in speed and reliability. In that manner, they would be ensuring a continuous security assessment with real-time vulnerability detection and automated remediation, thus minimizing the probability of security breaches (Pendyala, 2020). DevSecOps even fosters collaboration between development, operations, and security teams in breaking down silos and encouraging a type of common ownership of security (Tomas, Li, & Huang, 2019). In line with making the security posture better, editing the DevSecOps is also aligned with compliance and regulatory issues, making it especially suited for industries handling sensitive data like finance, healthcare, and government sectors (Morales et al., 2020). Security and agility can be accomplished by the combination of CI/CD with security to allow the resilience of software products towards the transformation of cyber threats while being relevant to fast-paced innovations in the market.

1.2. Importance of Integrating Security in DevOps

Security breaches and cyber threats have never been this fast-spreading, with attackers using advanced techniques such as zero-day exploitations, ransomware, and supply chain attacks to breach software systems. Security paradigms that adopt late-stage vulnerability testing and manual reviews are, therefore, ineffective in this rapidly changing environment (Heilmann, 2020). If security is only an afterthought at the end of the SDLC, then vulnerabilities remain unresolved during development when they can only be detected during expensive remediation phases. Furthermore, applying the fixes at the last minute will delay project schedules and hinder along with impacting software quality, leading to the erosion of customer trust and business continuity.

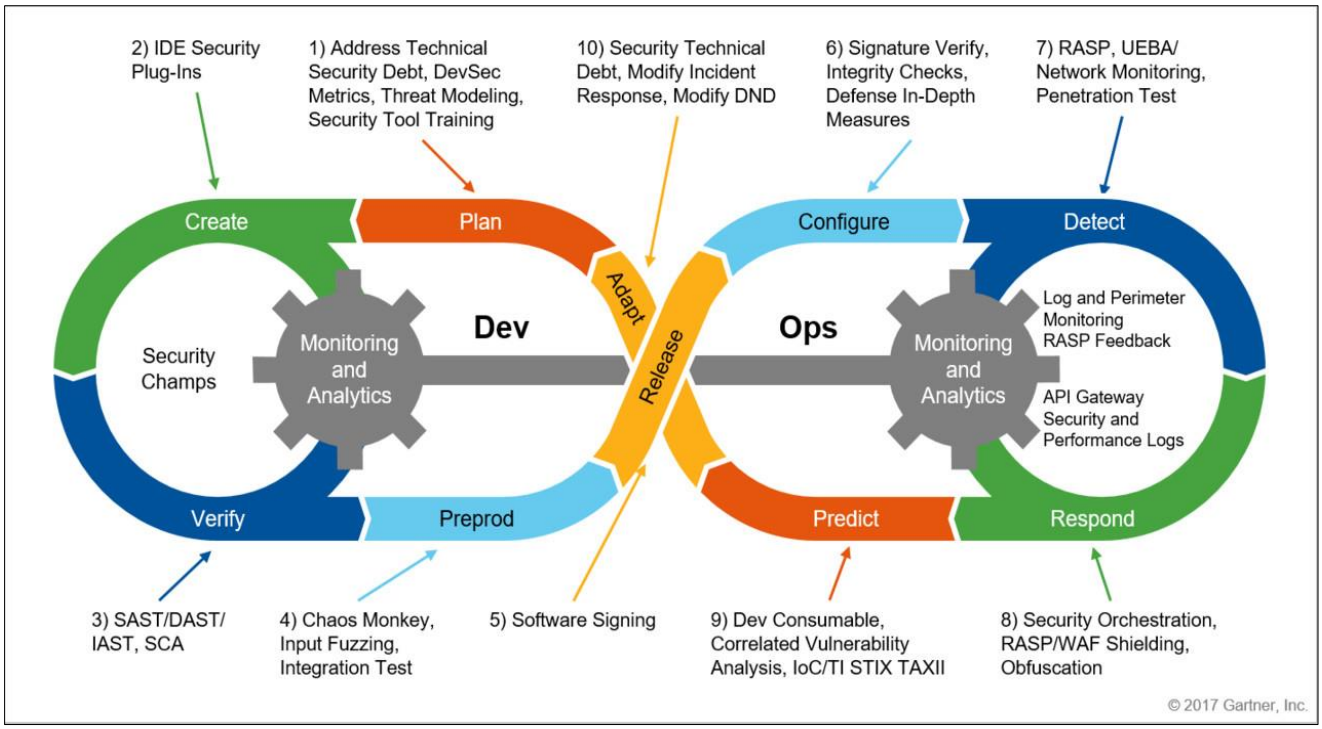

Figure 2 Importance of Integrating Security in Devops

Thus, the DevSecOps movement aims to avert this very situation via security throughout the entire development process, shifting security assessments to the left relative to the traditional SDLC timeline where security was an afterthought. This proactivity encompasses automated security tests, dynamic scans, static code analysis, security Linux operating systems, and compliance checks-well ahead of any deployment-(Prates et al., 2019). Integrating security into the DevOps pipeline enables organizations to implement continuous security validation without impeding organizational development velocity.

Perhaps the most significant advantage emanating from DevSecOps is faster detection and response to vulnerabilities. Real-time threat assessment and analysis are made possible through the accountability and validation of security; hence, any weakness from a security perspective can be reported way in advance before it escalates into a major incident (Mao et al., 2020). Thus, the land under continuous security testing is increasingly showing fast depreciation of the attack surface, thus guaranteeing software releases that are, besides being usable and functional, hard by design against potential cyber threats.

DevSecOps enhances collaborative culture of shared responsibilities amongst security teams, development teams, and operation team members, enhancing synergy among these teams. By contrast, siloed approaches have led to miscommunication in the past, resulting in prolonged remediation to fix security threats. Under DevSecOps, security becomes an integral part of the development workflow rather than being solely the domain of a specialized security team. Security professionals are integrated throughout the SDLC alongside development and operations to implement security policies and best practices via automation and IaC (Mao et al., 2020).

Having embedded security in its CI/CD pipeline enables organizations to balance speed of delivery and security, hence able to churn software faster with utmost security. It creates an enabling environment for the organization to bolster its security posture and agility in responding to market pressures.

Table 1 The Importance of DevSecOps in Modern Software Security

Aspects	Traditional Security Model	DevSecOps Approach
Timing of Security Integration	Late in the SDLC (post-development)	Early and continuous (shift-left approach)
Security Testing Methods	Manual reviews, late-stage vulnerability testing	Automated security testing, static and dynamic analysis
Risk of Vulnerabilities	High, due to undetected flaws persisting throughout development	Lower, as vulnerabilities are identified and mitigated early
Lower, as vulnerabilities are identified and mitigated early	Potential project delays and workflow disruptions	Seamless integration with DevOps pipeline
Collaboration Approach	Siloed security teams working separately	Shared responsibility across security, development, and operations

1.3. Research Problem and Motivation

It's DevSecOps worth the organizational effort, but really defining challenges from moving beyond a defensive mindset in development. Actually, what stands most in the way is cultural resistance to change within teams that have learned to work within their own traditional development and security workflows. Depending on different motivations, developers might be inclined towards speed and feature-set, while security personnel will usually enforce risk management and compliance. Most of the time interests clash and slow down collaboration for adding security in development pipelines (Morales et al., 2020). Such an attitude thus needs a paradigm shift inculcating a culture making security a collective responsibility rather than conceiving it as hindrance to fast development.

There is another challenge related to the introduction of varied levels of automation into the entire DevSecOps workflow as it is a common thing for the security tools to integrate seamlessly into CI/CD pipelines. Of course, selection of required security tools and trying to ensure compatibility with the development infrastructure for configuring automated security checks while minimizing false positives would surely spoil many organizations (Pendyala, 2020). Automating but not attended by sufficient consideration may introduce roadblocks to security while otherwise fastening software delivery.

Also open is the question of whether DevSecOps really brings any benefits in terms of software quality. DevSecOps claims that it is important by applying continuous security testing and vulnerability management principles; however, organizations struggle to come up with metrics to define and monitor that will prove really tangible effects. Empirically, there is evidence that assesses and quantifies the real impact of DevSecOps on software quality. Thus, it still remains a challenge for organizations to justify the cost and resources for implementation (Tomas, Li, & Huang, 2019). Metrics for evaluating this include defect density, MTTD (mean time to detect) vulnerabilities, MTTR (mean time to remediate), and software reliability, which warrant deeper analysis to form meaningful evidence of DevSecOps impact.

There are various skills gaps and resource limitations that organizations face while implementing DevSecOps. Secure coding forms, Infrastructure as Code (IaC), and security automation require specific skills that many development teams do not possess. Upskilling employees and hiring specialists in DevSecOps could be costly and time consuming, which also contributes to reduced adoption rates (Céspedes et al., 2020).

Overcoming those challenges included a systematic review and empirical assessment of software quality in regard to DevSecOps. The existing literature was synthesized, and real-world case studies were studied, through which this research measured the improvement of software security, efficiency, and reliability with DevSecOps.

1.4. Objectives of the Study

This research plans to determine how DevSecOps influences software quality through systematic literature review. Conduct an empirical study to assess the effectiveness of DevSecOps in real-world scenarios. Identify the major challenges in adopting DevSecOps and provide possible solutions. Offer recommendations for DevSecOps effectiveness to organizations.

1.5. Research Questions

The study is guided by the following research questions:

- How does DevSecOps influence software quality compared to traditional DevOps practices?
- What are the key challenges organizations face when adopting DevSecOps?
- What metrics can be used to measure the success of DevSecOps implementations?
- How can organizations overcome challenges in DevSecOps adoption?

1.6. Scope of the Study

A systematic literature review and empirical investigation will evaluate the influence of DevSecOps on software quality. The investigation will cover how security is introduced into the DevOps workflow, including automation tools, secure coding practices, and compliance frameworks. Since DevSecOps is believed to increase software quality, an assessment of primary quality attributes like security, reliability, maintainability, and performance is done. Identifying barriers to successful adoption, which include cultural resistance, complications concerning automation, and skills and knowledge gaps. Based on case studies, industrial reports, and experimental data from the real world that evaluate the benefits and challenges of applying DevSecOps in software development.

1.7. Significance of the Study

This research is important to both academia and industry because it addresses a gap in knowledge regarding the concrete advantages provided by DevSecOps concerning software quality. This study provides insights into evaluating how DevSecOps enhances security posture, minimizing vulnerabilities, maintaining compliance, and mitigating cyber threats. Many organizations find themselves confronted with cultural and technical barriers to working in a DevSecOps model. This research will suggest best practices, frameworks, and implementation strategies that ease the process of adoption. The research provides additional information into the impact of DevSecOps on development efficiency, security, and quality, contributing to the literature with data-driven insights.

2. Literature Review

DevSecOps becomes a natural evolution from DevOps by integrating security practices into every phase of the Software Development Lifecycle (SDLC). The idea is all about collaboration and automation, and on top of these principles there's continuous delivery-not just adding some security into already complete applications, just as Myrbakken and Colomo-Palacios (2017) take the view that DevSecOps refers to a structure that makes security be part and parcel of it as an ongoing and integrated process rather than just an afterthought. An automated security tool, compliance monitoring, and continuous testing help in detecting vulnerabilities early by reducing risk, which thus improves the integrity of software. Indeed, it has been in the addressing of security concerns not intrinsic to just DevOps practices that DevSecOps has evolved. Initially, DevOps was characterized by fast development and deployment without paying much attention to security considerations, resulting in an increasing number of vulnerabilities (Cui, n.d.). DevSecOps emanated partly in response to concerns of getting development pipeline security assessments to happen earlier rather than just the final few stages of the process (Mao et al., 2020). It is expected that increasing integration of proactive security within the workflow increases resilience in software while maintaining the efficiency of DevOps workflows.

DevSecOps has certain specific principles and practices that define it. One of the main principles is automation which is an enabler for all the security processes not to slow down the speed of development and deployment. There are static application security testing (SAST) and dynamic application security testing (DAST) tools that are used for automating security and identifying vulnerabilities early through such automation (Prates et al., 2019). Besides, compliance as code ensures that security policies are within the infrastructure, hence establishing a level playing field in enforcing equal security in all development processes (Heilmann, 2020). Another core practice is building a culture of security where developers, operations, and security teams share efforts without the silos that typically exist and experience a coherent security strategy (Morales et al., 2020).

In determining software quality in DevSecOps, it goes beyond merely functional performance to include and recognize security as an inherent quality attribute. Unlike conventional software quality measures, which mainly concerned themselves with reliability, maintainability, performance, and usability, security becomes an important aspect of quality in DevSecOps in order for the software to withstand threats and be as per standards in the industry's definition of quality (Céspedes et al., 2020). Security tests integrated into CI/CD pipelines continuously assess vulnerability, thus lowering risks and increasing overall robustness (Tomas, Li & Huang, 2019).

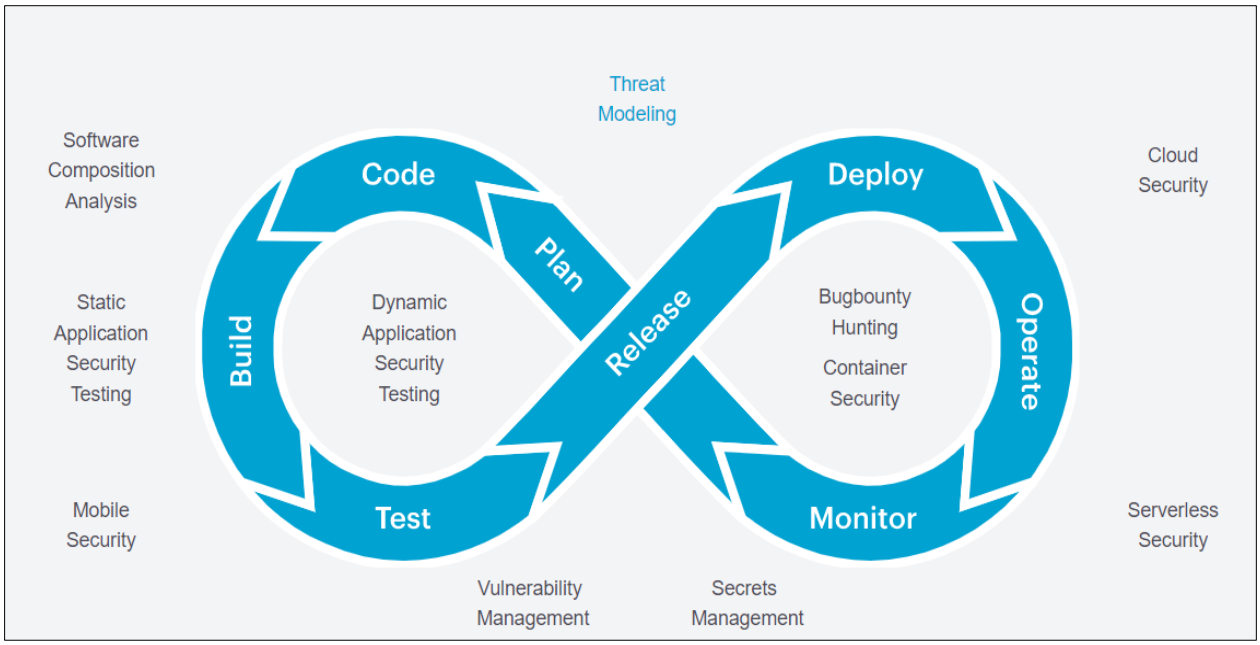

Figure 3 Software Quality and Security in Devsecops

Thus, the security-as-quality attribute in DevSecOps advocates risk management that's proactive. Indeed, contrary to conventional assessment processes carried out at the end of the SDLC, DevSecOps integrates security into the SDLC right from the very outset in the development stage, thus making it possible to address buy-in factors toward well-directed cost-effective kwags.

3. Methodology

The methodology for this research shall encompass systematic literature review as well as empirical evidence of the impact of DevSecOps on software quality. An automated review of literature in a structured format will help identify, analyze and correlate previous studies in the field vis-a-vis DevSecOps and software quality aspects. Relevant literature would be accessed through comprehensive search strategy using databases such as IEEE Xplore, SpringerLink, ACM Digital Library, and Google Scholar. Its SLA literature search incorporated keywords like "DevSecOps", "software quality", "security integration", and "continuous security". The following studies are considered according to the criteria prescribed for inclusion in the peer-reviewed and published archives from 2013 to 2023 regarding DevSecOps methodologies, security, and quality metrics of software. Excluded criteria contained duplicates, articles in languages other than English, and those that did not concern DevSecOps and software quality. In data extraction and synthesis, thematic of selected studies yields the key themes regarding implementation by DevSecOps, challenges that arise, benefits, and measurability with respect to software quality.

The empirical exercise is a mixed-method research design whereby qualitative and quantitative enhancement methods were used. Different data-gathering techniques were used-case studies, questionnaire surveys, or controlled experiments on organizations piloting DevSecOps. The purpose of case study analysis is to investigate industrial adoption, security practices, and actual challenges inherent to them. Survey data obtain the perspectives of software developers, security engineers, and IT managers regarding benefits they acquired from practicing DevSecOps. Controlled experiments assess how DevSecOps practices increase or improve software quality in terms of defect impact, security vulnerabilities, frequency of deployment, and mean time to recovery (MTTR) compared to environments without such practices. Participants chosen will be obtained from software development companies, financial institutions, and technology firms that are in transition to or planning to adopt DevSecOps.

Evaluated software quality metrics include the discovery of security vulnerabilities, defect density, review efficiency, and compliance, while the aforementioned tools that assisted in the analysis are hence evaluation for security and code quality: SonarQube, OWASP Dependency-Check, and other static code analysis frameworks. In particular, statistical analyses of survey responses, case studies comparison, and regression modeling will be carried out to correlate the introduction of DevSecOps with improvements in software quality. Finally, SLR and empirical study will be able to highlight and broaden understanding on how DevSecOps causes security assurance and improves software quality.

4. Findings and Discussions

The synthesis of findings from a systematic literature review has indicated that DevSecOps has managed to penetrate software development, with most of the researchers noting the integration of security practices within the DevOps workflow. Myrbakken and Colomo-Palacios (2017) note a paradigm shift from a continuous, integrated model of security to the regular legacy models of security. Common themes emerging from the literature include automation in security testing, cultural changes within development teams, and security-as-code adoption. A developing trend towards security metrics and assessment frameworks has emerged in numerous pieces of research such as observations made by Mao et al. (2020) and Prates et al. (2019). Unfortunately, there are still many gaps, especially in the standardization of DevSecOps implementations across industries. Measurement of software quality in a DevSecOps scenario would be inconsistent, as mentioned by Céspedes et al. (2020), and Kumar and Goyal (2020) also point out that effective automation of security processes is a big challenge.

From empirical studies, DevSecOps also turns out to impact software quality metrics. As found by Tomas, Li, and Huang (2019), organizations that are embracing DevSecOps practices are reporting improvements regarding the reliability of their software and their security posture. Mean Time to Detect (MTTD) and Mean Time to Remediate (MTTR) metrics have recorded an improvement since the shift to DevSecOps (Prates et al., 2019). Studies by Morales et al. (2020) and Heilmann (2020) also mention that DevSecOps leads to higher security compliance in said environments. Improvements in security, however, are usually a cost because, according to Sen (2021), there may be trade-offs while balancing speed and security in continuous integration and continuous deployment pipelines. The trade-off between the rapid delivery of software with slower but more rigorous security checks is, as Roche (2013) highlighted, an area of significant challenge.

The literature review and the evidence from empirical studies are converging towards the same point: they all recognize the benefits of DevSecOps but mark off some challenges. While the major thrust of the reviews in literature falls on theoretical frameworks and conceptual models (Cui, n.d.; Pendyala, 2020), empirical studies provide the real evidence that DevSecOps has improved outcomes in software quality and security. Either way, both sources have the same conclusion about the need for better-defined security metrics and more standardized methodologies for implementation. At the same time, Zhao, Clear, and Lal (n.d.) suggest that a global view on DevSecOps adoption is still lacking, indicating a need for more comprehensive research to capture diverse industrial and geographical contexts. To sum up, both literature and empirical findings support the premise that DevSecOps helps augment software security and quality but, on the contrary, standardization, automation, and speed-agile development of security requirements remain challenging ongoing research and industry areas.

Table 2 Showing Findings from Systematic Literature Review on DevSecOps

Key Theme	Literature Review Insights	Emprical Evidence
Security Integration	Shift from traditional security to continuous security integration	Improved security posture and reliability
Automation in Security Testing	Emphasis on security-as-code and automated security testing	Enhances security compliance in regulated environments
Security Metrics & Evaluation	Increasing focus on defining security metrics and evaluation frameworks	Metrics like MTTD and MTTR show improvement with DevSecOps
Standardization Challenges	Lack of uniform DevSecOps implementations across industries	Inconsistent measurement of software quality
Balancing Speed and Security	Challenges in maintaining rapid delivery while ensuring security	Trade-off between development speed and rigorous security checks
Global Adoption & Research Gaps	Need for a standardized, global perspective on DevSecOps adoption	More comprehensive research required across diverse industries and regions

5. Conclusion and Recommendations

5.1. Summary of Key Findings

Due to DevSecOps, an approach embedding security directly into software development, much advancement has occurred in the domain of software engineering. Unlike traditional strategies that would tackle security vulnerabilities at a later point in the software development life cycle, DevSecOps offers security at every stage, allowing for proactive management of risks and rapid mitigation of security threats. According to a systematic literature review and some empirical studies, DevSecOps provide improved software security through minimizing vulnerabilities, maximizing code stability, and building a security culture in the organization (Myrbakken & Colomo-Palacios, 2017; Heilmann, 2020). The other half of such a major paradigm shift is represented by automated security tools, permanent security monitoring, and the shift-left security paradigm: putting security testing early into the software development process, thus minimizing the risk of a security breach (Prates et al., 2019; Mao et al., 2020).

Complementarily, automation in DevSecOps significantly reduces the common bottleneck Security processes-vulnerability scanning, compliance checks, and threat detection. Furthermore, CI/CD pipelines supported by such automated security testing tools assist in real-time detection and remediation of security vulnerabilities, reducing the likelihood of security incidents after deployment. Security considerations, integrated within the DevOps model, have further encouraged developers to take greater accountability for security and therefore create more robust software (Céspedes et al., 2020).

The challenges are still faced at an organizational level even with its advantages. The technical difficulties and operational friction often arise when the security controls are considered too complicated to use or are not well integrated into their existing DevOps workflows (Tomas, Li, & Huang, 2019). The need for special skills in security, automation, and cloud technology creates a skills gap that forces organizations to invest in training and talent development to create security-minded development teams (Céspedes et al., 2020). Furthermore, members of DevSecOps will still face challenges in identifying if security is aligned with the speed of development, since sometimes security controls may obstruct the timely realization of software releases. Therefore, the successful application of DevSecOps depends on optimizing this area of security with rapid delivery (Morales et al., 2020).

Another important aspect in the adoption of DevSecOps is regulatory compliance. Nurturing the ecosystem of customers in highly regulated industries-such as finance, health, or government-compels assurance of compliance with security standards, as witnessed with GDPR, HIPAA, and ISO 27001, while still remaining agile in terms of the development process (Morales et al., 2020). Putting compliance checks within the automated security framework allows companies to meet regulatory obligations without compromising development speed.

5.2. Implications for Software Development Teams and Organizations

The successful implementation of DevSecOps requires a structured and strategic approach to bring it in line with the requirements of the organization in question, which moves beyond pure tool adoption. Companies should concentrate on the three core areas: workforce training, appropriate tool selection, and cultural change. Bridging the training gap is critical because, in many cases, there are significant differences between the striker-grown antiquities of development and security teams. Most developers lack any specialized knowledge related to security, while their counterparts in security have not really understood the methodologies of contemporary software development. An effective answer to such challenges requires continuous learning programs with a focus on security-related training, hands-on workshops, or certifications addressing principles to be followed in DevSecOps. This will raise awareness and facilitate the inculcation of necessary skills that will contribute to creating a development environment that is most cognizant of security as a shared responsibility instead of a separate line stage in the software development life cycle (Pendyala, 2020).

Denoting the fact that they must adopt certain best practices, apart from training, to smooth out the edges in the joining of security in DevOps workflows. Continuous security testing makes it possible to detect and fix security flaws before introducing the military might of production, thus preventing any propagation of security defects into production. Automated compliance checks further enhance security by lowering the chances of errors made by humans and ensuring the most efficient receipt of regulatory requirements. Automation tools are very important because they help keep security intact while the development speed does not get killed-off. Security testing should also be incorporated into the continuous integration and continuous deployment (CI/CD) pipelines, allowing the organizations to integrate into their software development workflows a form of embedded security without bottlenecks (Kumar & Goyal, 2020; Hsu, 2018).

Other important aspects of successful DevSecOps include fostering collaboration within the triad-delineating development, security, and operations. Meanwhile, conventional software development paradigms create silos among the respective teams, creating miscommunication and inefficiencies that slow down security efforts. DevSecOps points out the need for shared responsibility and close collaboration in deciding what security measures will be integrated early on in the entire software development lifecycle. Such an argument advocates for open communications and cross-functional teaming arrangements toward the effective implementation of security measures such that security becomes a continuous process and not an afterthought.

Regulatory compliance will also be another factor when considering implementation of DevSecOps by an organization. Since different industries are governed by different security and compliance requirements, there are some common financial regulations such as Payment Card Industry Data Security Standard (PCI DSS). The field of healthcare also has specific mandates, for instance, with regard to the Health Insurance Portability and Accountability.

5.3. Future Research Directions

Therefore, future research needs to be oriented toward refining the metrics for DevSecOps evaluation to better assess their long-term impacts on improving software security, development efficiency, and overall organizational performance. Existing studies present short-term benefits, such as reduction of vulnerabilities and an increased fraction of secure compliance; however, they do not present comprehensive long-term metrics measuring the effectiveness of DevSecOps. Hence, standardizing frameworks for measuring security improvements, development speed, and operational resiliency will also pave the way for enabling organizations to make data-driven decisions when adopting DevSecOps methodologies (Prates et al., 2019).

Another critical aspect for research could be anti-threatening capabilities automated with AI. It could provide potentially groundbreaking systems in the detection, mitigation, and management of vulnerabilities and/or threats. Machine learning algorithms have some security processes related to their functionality: anomaly detection, incident response automation, and predictive threat modeling. In fact, efficiency and effectiveness could be increased by applying AI technological tools in security testing, minimizing the number of false positives in vulnerability assessments, such as real time threat intelligence sharing. As cyber threats will evolve with time, the need for AI solutions integrated into DevSecOps workflows will become inevitable toward continued proactive security posture (Cui, n.d.).

Cross-sectoral studies will also yield useful reflections concerning the country-specific demands and practices germane to DevSecOps adoption. For example, security requirements for financial institutions will be deeply different from that of healthcare or governmental agencies. Information about how organizations in different sectors have managed to adopt DevSecOps, how they reformulate their security policies, and how they have gone about complying with regulations can provide an avenue for refining DevSecOps to address an industry's specific needs. Research doing analysis across industries' study domains will enable a more fine-grained adaptation of strategies for DevSecOps frameworks to meet sector-specific security requirements (Sen, 2021; Zhao, Clear, & Lal, n.d.).

Beyond that, there is also ample opportunity for research into future technologies in relation to diverting 'normal' operations in DevSecOps such as blockchain and zero-trust architectures. Blockchain is decentralized and a tamper-resistant, immutable logs can be used for tracking changes in code repositories, assuring software integrity, and ensuring transparency in supply chain security and thus augments the security perspective on software. Zero-trust architectures, based on the principle of continuous authentication and least-privilege access, must also strengthen controls in DevSecOps by minimizing exposure to insider threat and unauthorized access for the DevSecOps environment. This synergy between these technologies and DevSecOps will lead to robust best practices surrounding the resilience of software development, securing it in every stage of development lifecycle.

References

[1] Challenges and solutions when adopting Myrbakken, H., & Colomo-Palacios, R. (2017). DevSecOps: a multivocal literature review. In Software Process Improvement and Capability Determination: 17th International Conference, SPICE 2017, Palma de Mallorca, Spain, October 4–5, 2017, Proceedings (pp. 17-29). Springer International Publishing.

[2] Myrbakken, H., & Colomo-Palacios, R. (2017). DevSecOps: a multivocal literature review. In Software Process Improvement and Capability Determination: 17th International Conference, SPICE 2017, Palma de Mallorca, Spain, October 4–5, 2017, Proceedings (pp. 17-29). Springer International Publishing.

[3] Prates, L., Faustino, J., Silva, M., & Pereira, R. (2019). Devsecops metrics. In Information Systems: Research, Development, Applications, Education: 12th SIGSAND/PLAIS EuroSymposium 2019, Gdansk, Poland, September 19, 2019, Proceedings 12 (pp. 77-90). Springer International Publishing.

[4] Mao, R., Zhang, H., Dai, Q., Huang, H., Rong, G., Shen, H., ... & Lu, K. (2020, December). Preliminary findings about devsecops from grey literature. In 2020 IEEE 20th international conference on software quality, reliability and security (QRS) (pp. 450-457). IEEE.

[5] Heilmann, J. (2020). Application Security Review Criteria for DevSecOps Processes.

[6] Céspedes, D., Angeleri, P., Melendez, K., & Dávila, A. (2020). Software product quality in DevOps contexts: a systematic literature review. In Trends and Applications in Software Engineering: Proceedings of the 8th International Conference on Software Process Improvement (CIMPS 2019) (pp. 51-64). Springer International Publishing.

[7] Tomas, N., Li, J., & Huang, H. (2019, June). An empirical study on culture, automation, measurement, and sharing of devsecops. In 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security) (pp. 1-8). IEEE.

[8] Cui, J. Comparative Analysis of DevOps, DevSecOps, and AIOps in Software Development Architectures: Impacts on Software Quality and Development Efficiency.

[9] Pendyala, V. (2020). Evolution of integration, build, test, and release engineering into devops and to DevSecOps. In Tools and techniques for software development in large organizations: emerging research and opportunities (pp. 1-20). IGI Global.

[10] Roche, J. (2013). Adopting DevOps practices in quality assurance. Communications of the ACM, 56(11), 38-43.

[11] Hsu, T. H. C. (2018). Hands-On Security in DevOps: Ensure continuous security, deployment, and delivery with DevSecOps. Packt Publishing Ltd.

[12] Morales, J. A., Scanlon, T. P., Volkmann, A., Yankel, J., & Yasar, H. (2020, August). Security impacts of sub-optimal DevSecOps implementations in a highly regulated environment. In Proceedings of the 15th International Conference on Availability, Reliability and Security (pp. 1-8).

[13] Kumar, R., & Goyal, R. (2020). Modeling continuous security: A conceptual model for automated DevSecOps using open-source software over cloud (ADOC). Computers & Security, 97, 101967.

[14] Sen, A. (2021). DevOps, DevSecOps, AIOPS-paradigms to IT operations. In Evolving Technologies for Computing, Communication and Smart World: Proceedings of ETCCS 2020 (pp. 211-221). Springer Singapore.

[15] Hong, J. K. (2019). Component analysis of DevOps and DevSecOps. Journal of The Korea Convergence Society, 10(9), 47-53.

[16] Zhao, X., Clear, T., & Lal, R. Defining Devsecops and the Missing Global Dimension: A Multi-Vocal Literature Review. Available at SSRN 4511782.

[17] Horvath, M. (2018, March 8). Integrating Security into DevOps: Adapting the tools and techniques you already have to a faster pace of iterative development. https://www.linkedin.com/pulse/integrating-security-devops-adapting-tools-techniques-mark-horvath